Microsoft, .NET and Eiffel

Elliott McCrory
6 November 2000

Eiffel Power
from ISE

Microsoft

Microsoft .net

msdn online

Summary Slide

- Observations on Software Engineering
- Bertrand Meyer
- .NET
  - Environment, Languages, etc.
  - What is it?
- Eiffel
  - Key Features
  - Examples

Observations on Software Engineering

1. Object-oriented (OO) technologies are here to stay
2. Many languages support OO
   - Java, C++, Ada, Visual Basic, …
3. OO Operating Systems
   - Java VM, CORBA, COM, EJB, …
4. Microsoft only contributes to things it can own.

Bertrand Meyer

- Professor at Monash Univ., AU
- Chief egghead for ISE, San Diego, CA
- Prolific author on OO ideas
  - "Object Oriented Software Construction"
  - "Object Success: A Manager's Guide to Object Orientation, …"
  - *Amazon.com: 33 other titles*
- I attended two seminars hosted by him
  - Oct 1998: "Software Design By Contract"
  - Oct 2000: ".NET in One Day"

Microsoft .net msdn online

- An OO operating environment
- Very similar concept to Java
  - MS can’t own Java, therefore …
- Relevant Technical Details …

.NET: Environment

- Heart: OO “Assembly Language”
  - Instead of the Java ByteCode
- Many architectures will support this OOAS
  - Windows 2000
  - Successor to Windows ME
  - Successor to Windows CE
  - Others???
- Replaces Windows runtime environment
  - No more registry or DLL’s

.NET: Languages

- Multiple language support
  - C#
  - Eiffel, C++ (*sorta*), COBOL (!), SmallTalk
  - Perl, Python, APL
  - A host of academic languages
    - CAML, Mercury, Scheme, Oberon, Component Pascal
- Cross-language interoperability
  - Method calls
  - Associations
  - Debugging!
  - Inheritance!!

.NET: Miscellaneous

- Managed in the assembly:
  - Garbage Collection
    - *Comments on Managed vs. unmanaged C++*
  - Security
  - Threads
  - Debugging
- Native code can be executed
  - But not "managed"
- Significant "Metadata"
- Serialization handled automatically
  - Several mechanisms, including XML/SOAP

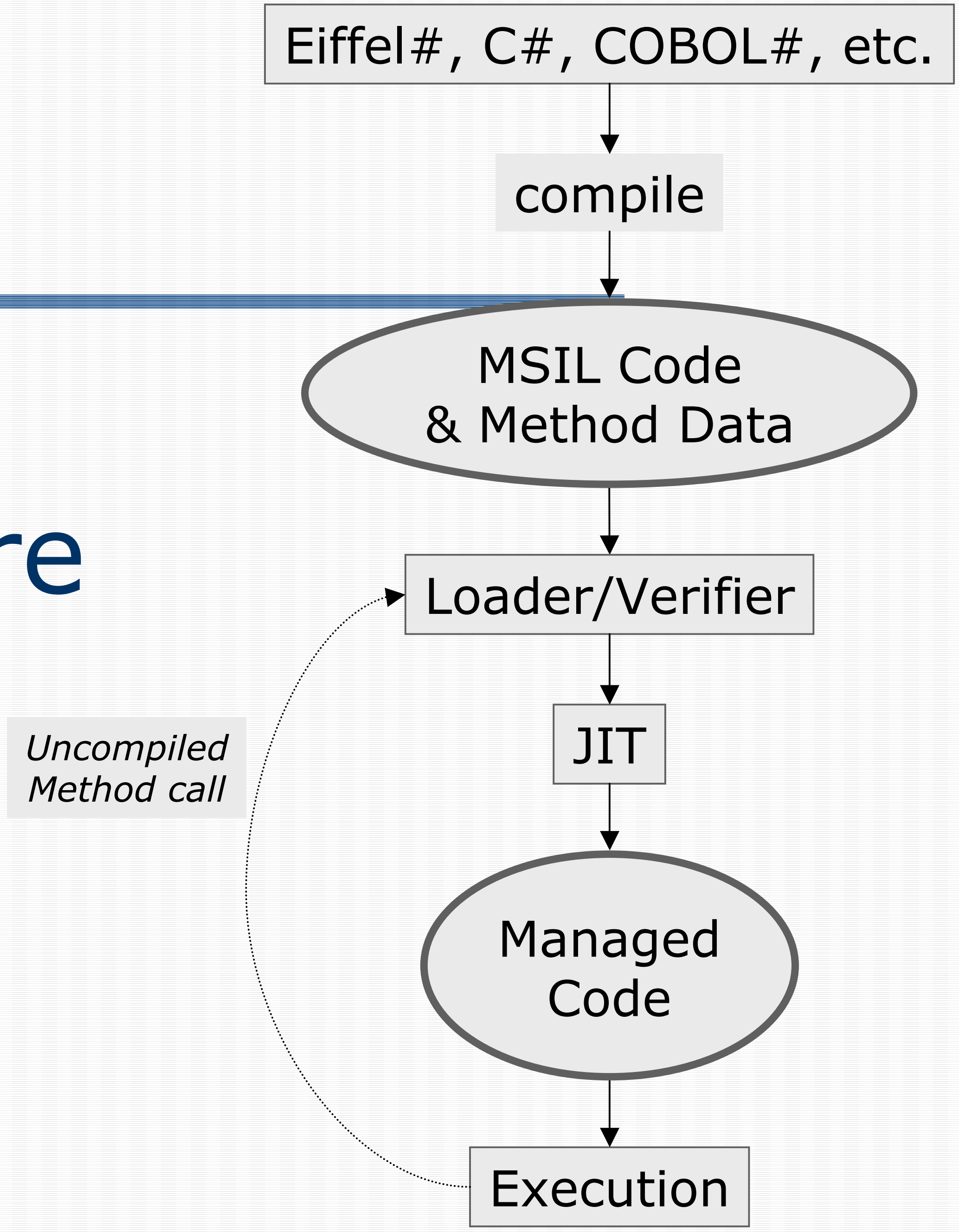
.NET: What is it? *(Meyer)*

- A virtual machine above the OS
- A language interoperability architecture
- A common runtime for many languages
- An architecture for Internet and Web development
- A component model, replacing COM
- A standardized versioning mechanism
- A uniform security policy
- Thousands of reusable components
- An interoperability standard – SOAP
- The cornerstone of Microsoft's future development

.NET: What is it? *(McCrory)*

- Microsoft’s version of Java
- A more general operating environment than the Java VM
- A replacement for COM, DCOM, etc.
- A chance that the “Windows Environment” will expand beyond Microsoft
  - *Fat chance!*

.NET Architecture



Eiffel

Eiffel Power

from ISE

- An elegant OO language

  From eiffel.com:

  What is eiffel? Answering "an object-oriented language" is correct, but only part of the story. Eiffel is the only O-O language that also includes a comprehensive approach to software construction: a method, and an environment (ISE Eiffel). The language itself is not just a programming language but also covers analysis, design and implementation.

- Eiffel# is the .NET version
  - Had to restrict it
    - .NET does not support multiple inheritance, but Eiffel does.

Eiffel: Key Features

- Assertions: "Design by Contract®"
  - Preconditions, post conditions, invariants
  - Leads to "Short Form" documentation
- Genericity
  - True general classes
  - Not templates—part of the language
- Other syntax features
  - Multiple inheritance, even through genericity
  - Argument-less methods
    - Look like "attributes"
    - `field := kludge * quadInstance.strength;`

Eiffel Example (*canned*)

```
class ACCOUNT creation
    make
feature

  balance : INTEGER;
  owner : PERSON;
  minimum_balance : INTEGER is 1000;

  open(who: PERSON) is
        -- Assign the acct person "who"
  do
    owner := person;
  end; -- open

  deposit(sum: INTEGER) is
  require
    sum >= 0;
  do
    add(sum);
  ensure
    balance = old balance + sum
  end; -- deposit

withdraw(sum: INTEGER) is
  require
    sum >= 0;
    sum <= balance - minimum_balance
  do
    add(-sum);
  ensure
    balance = old balance - sum
  end; -- withdraw
feature { NONE }

  add (sum: INTEGER) is
  do
    balance := balance + sum
  end; -- add

  make (initial: INTEGER) is
  require
    initial >= minimum_balance
  do
    balance := initial
  end; -- make

invariant
  balance >= minimum_balance
End; -- class ACCOUNT
```

Eiffel Short Form

```
class ACCOUNT creation
    make
feature

  balance : INTEGER;
  owner : PERSON;
  minimum_balance : INTEGER;

  open(who: PERSON);
  deposit(sum: INTEGER)
  require
    sum >= 0;
ensure
    balance = old balance + sum
  end; -- deposit
withdraw(sum: INTEGER)
  require
    sum >= 0;
    sum <= balance - minimum_balance
ensure
    balance = old balance - sum
  end; -- withdraw
invariant
  balance >= minimum_balance
End; -- class ACCOUNT
```

Made-up Example

```
class CIRCUIT
feature
  setCurrent(value: DOUBLE) is
        -- Set the current in this circuit
  require
    value >= min;
    value < max
  do
    current := value;
  end; -- setCurrent

  make (m1: DOUBLE, m2: DOUBLE) is
  require
    m1 < m2;
  do
    min := m1;
    max := m2
  end; -- make

feature { NONE  }
  current : DOUBLE; -- The current in the circuit
  min : DOUBLE; -- Minimum allowable current
  max : DOUBLE; -- Maximum allowable current

invariant
  current >= min;
  current < max
end -- class ACCOUNT
```

```
class MAGNET_CIRCUIT inherit
    CIRCUIT;
feature

  I2S : DOUBLE; -- Conversion from Current to Strength

  setI2S(v: DOUBLE) is
  do
   I2S := v
  end; -- setI2S

  strength is
  do
    Result := I2S * current
  end; -- strength

end; -- class MAGNET_CIRCUIT
```